Proceedings, Komputer dan Sistem Intelijen (KOMMIT2004)
Auditorium Universitas Gunadarma, Jakarta, 24 – 25 Agustus 2004 ISSN : 1411-6286

UJI PRIMALITAS TEOREMA POCKLINGTON DENGAN MENGGUNAKAN PROGRAM KOMPUTER *)

Ngarap Im Manik [1], Sangadji [2]
Andy Sumantri Harsono [3]

1) & 2) Staf Pengajar FMIPA-UBINUS, Jakarta e-mail manik@binus.ac.id
3) Alumni FMIPA-UBINUS, Jakarta

Abstrak

Bilangan Prima merupakan salah satu bagian dari teori bilangan yang merupakan konsep dasar dan utama dari ilmu Matematika. Banyak penelitian telah dilakukan untuk mengembangkan teori bilangan prima seperti pencarian dan pengujian bilangan prima yang semuanya dilakukan dengan perhitungan secara manual. Sehubungan dengan hal di atas dalam makalah ini mencoba untuk melakukan pengujian bilangan prima dengan menggunakan program komputer melalui perancangan algoritma program berdasarkan teorema Pocklington.

Setelah dilakukan pengujian program, diperoleh hasil evaluasi bahwa program yang dibuat dapat menguji bilangan-bilangan prima. Kekurangan yang diperoleh bahwa pengujian bilangan prima yang jumlah digitnya semakin besar, maka lama waktu proses juga semakin bertambah.

Kata Kunci : *Bilangan Prima, Uji Primalitas, Teorema Pocklington, Faktorisasi.*

1. Pendahuluan

Bilangan adalah suatu bagian yang tak terpisahkan dari kehidupan kita, dan telah dikenal semua orang sejak kecil. Teori bilangan telah menempati posisinya yang unik dalam dunia matematika. Hal ini disebabkan oleh subjek – subjek berkaitan yang selama ini tidak dapat dipertanyakan maupun dijawab. Tetapi sekarang dapat disaksikan penemuan – penemuan yg baru dan menakjubkan dari sifat – sifat bilangan. Dan pada suatu bagian dari karir mereka, banyak dari ahli matematika telah menyumbangkan kepandaian mereka pada ilmu ini.

Salah satu bagian dari teori bilangan yang sangat penting dan mendukung setiap aspek dari teori bilangan itu sendiri adalah pemikiran mengenai teori bilangan prima. Teori bilangan prima menjadi begitu penting karena setiap bilangan bulat positif yang lebih besar dari 1 adalah prima atau hasil kali dari bilangan - bilangan prima dengan penyajiannya atau penulisannya yang tunggal terlepas dari urutan faktor-faktornya.

Penulis ingin mengangkat masalah uji primalitas karena sekarang ini uji primalitas merupakan suatu subjek cukup baru dan masih terus dikembangkan. Kemajuan yang cukup besar dalam teknologi komputerisasi telah meningkatkan ketertarikan orang untuk melakukan perhitungan – perhitungan yang besar, yang kemudian mendatangkan pengembangan algoritma – algoritma baru untuk dapat mengenali bilangan prima secara cepat. Sebagian dari algoritma ini membutuhkan kemampuan perhitungan komputer yang cukup cepat sehingga algoritma ini tidak dapat diproses pada jaman dahulu. Dukungan algoritma ini terhadap cepatnya jalur dan sistem keamanan informasi sangatlah besar.

Tulisan ini dibatasi pada uji bilangan bulat positif n, di mana $1 < n < 1.000.000.000$. serta program komputer yang dirancang menggunakan bahasa pemrograman Pascal.

2. Landasan Teori

Pada penelitian dan perancangan algoritma ini , akan dibahas mengenai beberapa teorema uji primalitas yang telah ditemukan baru – baru ini. Teorema – teorema ini tidak seluruhnya akan

digunakan dalam pengembangan algoritma, oleh karena keterbatasan dari masing – masing teorema. Berikut ini adalah teorema – teorema uji primalitas.

Definisi 1 (Bilangan Prima)
Suatu bilangan bulat positif p yang lebih besar dari 1 disebut *bilangan prima*, jika pembagi positifnya hanyalah 1 dan p. Bilangan bulat positif yang lebih besar dari 1 dan tidak prima disebut sebagai bilangan *komposit*. Dari definisi ini jelas bahwa 1 bukan bilangan prima meskipun pembagi positif dari 1 hanyalah dia sendiri. Juga jelas bahwa satu - satunya bilangan prima yang genap adalah bilangan 2.

Teorema 1. Teorema Lucas
Jika terdapat suatu bilangan bulat a di mana $a^{n-1} \equiv 1 \pmod{n}$ dan $a^{(n-1)/p} \equiv 1 \pmod{n}$ untuk semua bilangan prima p yang habis membagi n-1, maka n adalah bilangan prima.
Bukti
Misalkan a mempunyai order k modulo n. Teorema 2.8.1 menyatakan bahwa kondisi $a^{n-1} \equiv 1 (\mathrm{mod}\, n)$ mengakibatkan $k \mid n-1$; katakanlah, $n-1 = kj$ untuk suatu j. Jika $j > 1$, maka j mempunyai suatu pembagi prima q. Sehingga terdapatlah suatu bilangan bulat h yang memenuhi $j = qh$. Sebagai hasilnya, $a^{(n-1)/q} = (a^k)^h \equiv 1^h \equiv 1 (\mathrm{mod}\, n)$
yang bertentangan dengan hipotesis di atas. Sehingga diperoleh $j = 1$. Tapi kita tahu bahwa order dari a tidak melebihi $\phi(n)$. Jadi, $n-1 = k \le \phi(n) \le n-1$, sehingga $\phi(n) = n-1$, di mana hal ini menyatakan bahwa n adalah prima.
Contoh
Kita ambil $n = 997$. Untuk $a = 7$, diperoleh $7^{996} \equiv 1 (\mathrm{mod}\, 997)$. Karena $n-1 = 996 = 2^2 \cdot 3 \cdot 83$, kita hitung $7^{996/2} = 7^{498} = -1 (\mathrm{mod}\, 997)$ Á $7^{996/3} = 7^{332} = 304 (\mathrm{mod}\, 997)$
$7^{996/83} = 7^{12} = 9 (\mathrm{mod}\, 997)$. Berdasarkan Teorema 1, dapat dibuat bahwa 997 adalah prima.

Teorema 2. Pengembangan Teorema Lucas
Jika untuk setiap bilangan prima p_i yang membagi $n-1$ terdapatlah suatu bilangan bulat a_i di mana $a_i^{n-1} \equiv 1 \pmod{n}$ tetapi $a_i^{(n-1)/pi} \equiv 1 \pmod{n}$, maka n adalah bilangan prima.
Bukti
Misalkan bahwa $n-1 = p_1^{k_1} p_2^{k_2} \cdots p_n^{k_n}$, di mana para p_i merupakan bilangan – bilangan prima yang berlainan. Misalkan juga h_i adalah order a_i modulo n. Berdasarkan fakta bahwa $h_i \mid n-1$ dan $h_i \mid (n-1)/p_i$ menghasilkan $p_i^{ki} \mid h_i$. Tetapi untuk setiap i kita dapatkan $h_i \mid \phi(n)$, sehingga $p_i^{ki} \mid \phi(n)$. Akibatnya $n-1 \mid \phi(n)$, sehingga n bilangan prima.
Contoh
Kita ambil lagi $n = 997$. Dengan mengetahui bahwa pembagi – pembagi prima dari $n-1 = 996$ adalah 2, 3 dan 83, kita peroleh untuk basis – basis yang berbeda 3, 5 dan 7 bahwa

$$3^{996/83} = 3^{12} = 40 (\mathrm{mod}\, 997)$$
$$5^{996/2} = 5^{498} = 996 (\mathrm{mod}\, 997)$$
$$7^{996/3} = 7^{332} = 304 (\mathrm{mod}\, 997).$$

Sesuai dengan Teorema 2, maka dapat kita simpulkan bahwa 997 adalah prima.

Kedua teorema di atas sudah cukup mengurangi permasalahan uji primalitas bilangan n dengan cara menggunakan faktorisasi dari $n-1$. Tetapi untuk mencari faktor – faktor dari $n-1$ tidak lebih mudah daripada mencari faktor – faktor dari bilangan n itu sendiri. Selain itu pencarian pengujian memerlukan pemangkatan dari suatu basis terhadap bilangan itu sendiri dibagi masing – masing faktor dari bilangan tersebut. Semakin banyak faktor dari bilangan tersebut perhitungan yang dibutuhkan untuk pengujian semakin banyak, sehingga perhitungan menjadi semakin rumit dan panjang. Oleh karena itu dikembanganlah teorema berikut.

Teorema 3. Teorema Pocklington

Misalkan $n-1 = mj$, di mana $m = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$, $m \geq \sqrt{n}$ dan $\gcd(m,j) = 1$. Jika untuk setiap bilangan prima p_i $(1 \leq i \leq s)$ terdapatlah suatu bilangan bulat a_i di mana $a_i^{n-1} \equiv 1 (\mathrm{mod}\, n)$ dan $\gcd(a_n^{(n-1)/p_i} - 1, n) = 1$, maka n adalah bilangan prima.

Bukti

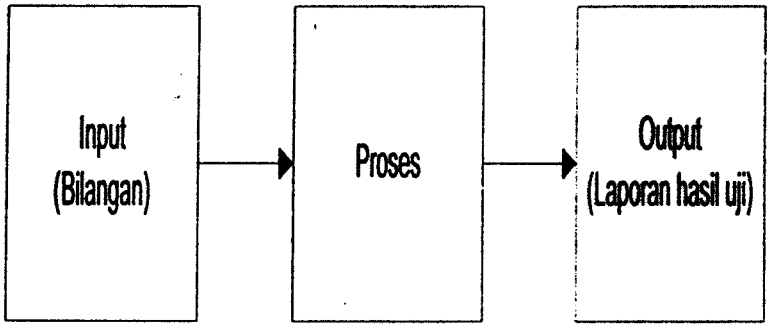
Tidak berbeda jauh dengan argumen kita pada Teorema 3.1.2. Misalkan p sebagai pembagi prima dari n dan kita ambil h_i sebagai order a_i modulo p, sehingga $h_i \mid p-1$. Dari kongruensi $a_i^{n-1} \equiv 1 (\mathrm{mod}\, p)$, kita dapatkan $h_i \mid n-1$. Hipotesis $\gcd(a_i^{(n-1)p_i} - 1, n) = 1$ menyatakan bahwa $a_i^{(n-1)p_i} \neq 1 (\mathrm{mod}\, p)$ sehingga $h_i \mid (n-1)/p_i$. Dapat diperlihatkan bahwa $p_i^{k_i} \mid h_i$, yang menghasilkan kita pada $p_i^{k_i} \mid p-1$. Karena hasil ini berlaku untuk setiap i, maka $m \mid p-1$. Sehingga kita dapatkan kontradiksi bahwa ada pembagi prima dari n yg lebih besar dari $m \geq \sqrt{n}$. Jadi n adalah prima.

Dari teorema di atas, faktor – faktor yang diperlukan dalam pengujian n hanyalah sebatas faktor – faktor yang tergabung dalam m. Shg proses perhitungan jauh lebih singkat.

3. Rancangan Program Komputer

Struktur Program

Secara umum program komputer dengan bahasa pemrograman pascal yang dibuat terdiri dari input, proses dan output, yang dapat ditunjukkan pada gambar berikut:



Gambar 1. Struktur Program

Input yang diperlukan program berupa bilangan. Di mana bilangan ini merupakan bilangan ganjil yang lebih besar dari 2. Hasil / keluaran dari program adalah laporan yang menyatakan bahwa bilangan masukan merupakan bilangan prima atau bukan dengan menyertakan salah satu faktor yang menyatakan bahwa bilangan tersebut bukan bilangan prima. Pengujian berdasarkan teorema uji primalitas Pocklington. Tahap pengujian ada 2, yaitu pengujian berdasarkan uji modulo dan pencarian gcd bilangan tersebut dengan bilangan uji.

Tipe Data yang digunakan

Untuk dapat menguji bilangan yang cukup besar diperlukan penampung data (tipe data) yang besar pula. Pada metode yang penulis gunakan, terdapat suatu pengujian berdasarkan rumus $\gcd(a_n^{(n-1)/p_i} - 1, n) = 1$, di mana n sangat menentukan besar $a_n^{(n-1)/p_i}$ yang akan ditampung. Nilai $a_n^{(n-1)/p_i}$ jauh lebih besar dari daya tampung tipe data yang telah tersedia, sehingga hasil perhitungan menjadi tidak akurat. Oleh karena itu diperlukanlah suatu tipe data baru yang dapat menampung hasil perhitungan di atas dan dukungan algoritma – algoritma perhitungan untuk tipe data yang baru tersebut.

Tipe data baru yang akan penulis gunakan adalah tipe data *String*, di mana tipe data ini dapat menampung bilangan yang sangat besar. Supaya tipe data ini dapat kita gunakan dalam perhitungan, maka diperlukanlah modul – modul untuk mengoperasikan tipe data tersebut. Modul dasar operasinya adalah Modul Penjumlahan, Modul Pengurangan, Modul Perkalian dan Modul Pembagian. Kemudian modul – modul ini digunakan sebagai dasar perhitungan dari fungsi – fungsi / modul – modul yang lain. Modul Penjumlahan dan Modul Pengurangan sama seperti dasar algoritma penjumlahan dan pengurangan bilangan. Modul Perkalian dari bilangan a dan b menggunakan Modul Penjumlahan bilangan a diulang sebanyak b kali. Sedangkan Modul Pembagian (a membagi b) menggunakan Modul Pengurangan b dengan a terus menerus hingga b lebih kecil dari a.

Jumlah perulangan dari pengurangan terserbut adalah hasil bagi, dan nilai b terakhir yang lebih kecil dari a merupakan sisa.

Tahapan Proses Perhitungan

1) Tahap pencarian GCD

Sesuai namanya, modul ini bertujuan untuk mencari gcd dari 2 buah bilangan. Proses perhitungannya berdasarkan teorema – teorema GCD, khususnya algoritma pembagian. Proses pencarian gcd berdasarkan modul pembagian, di mana bilangan yang lebih besar dibagi bilangan yang lebih kecil sehingga didapatlah sisa bagi. Kemudian bilangan lebih kecil digunakan dalam proses pembagian oleh sisa tersebut sehingga didapat sisa yang baru. Proses ini berulang terus hingga sisa yang didapat sama dengan nol atau satu, maka didapatlah nilai gcd dari 2 bilangan tersebut.

2)Tahap Utama Uji Primalitas

Terdapat 3 tahap pengujian,yaitu tahap pencarian faktor $n-1$, dan 2 tahap pengujian.

Tahap pertama dari Modul Uji Primalitas adalah mencari faktor dari $n-1$, di mana n adalah bilangan yang diinput / akan diuji. Pemfaktoran $n-1$ berdasarkan Teorema Fundamental Aritmetika.dan Metode Eratosthenes, di mana nilai $n-1$ diuraikan satu – persatu dimulai dari bilangan prima terkecil, yaitu 2, dengan menggunakan Modul Pembagian sampai dengan $\sqrt{n}$, hingga didapatlah faktor – faktor primanya. Kemudian dari faktor – faktor prima tersebut didapatkanlah suatu variabel m dan j, di mana $p = m \cdot j$, $m > \sqrt{n}$ dan $m = p_1^{k_1} p_2^{k_2} \cdots p_s^{ks}$. Untuk suatu $n = a / b$, di mana $a \geq b$, nilai n akan semakin kecil jika nilai b semakin besar. Sehingga untuk memperkecil proses perhitungan, nilai m sebagai pembagi dari $(n-1)$, sebaiknya diperoleh dari perkalian faktor – faktor prima $n-1$ yang terbesar, hingga memenuhi syarat m > $\sqrt{n}$.

Tahap kedua dari Modul Uji Primalitas adalah pengujian dari faktor – faktor $n-1$ terhadap n dengan rumus $a_i^{n-1} \equiv 1 (\mathrm{mod}\, n)$ untuk sebarang basis a_i. Basis a_i merupakan bagian dari faktor – faktor p_i. Untuk mempermudah pengujian maka sebaiknya kita gunakan basis a_i yang relatif kecil, misal $a_i = 2$. Sehingga hasil pemangkatan a_i tidak terlalu besar, sehingga proses perhitungan dapat memakan waktu yang lebih cepat.

Perhitungan $a_i^{n-1} \equiv 1 (\mathrm{mod}\, n)$ dapat dipermudah dengan menggunakan Teorema Fermat. Jika $a_i^{n-1} \equiv 1 (\mathrm{mod}\, n)$ telah terpenuhi maka pengujian tahap ketiga dapat dilakukan.

Tahap ketiga dari Modul Uji Primalitas adalah pengujian terakhir berdasarkan rumus $\gcd(a_n^{(n-1)/p_i} - 1, n) = 1$. Pengujian ini adalah perhitungan yang paling sulit untuk dilakukan oleh karena besarnya bilangan yang akan diuji akan sangat besar, yaitu $a_n^{(n-1)/p_i}$, sehingga perlu waktu perhitungan yang relatif lama.

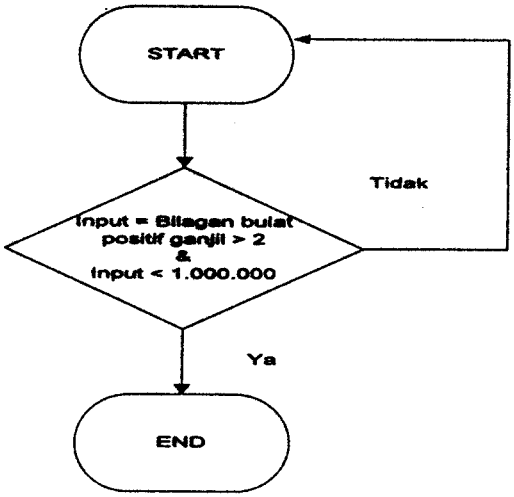
Jika bilangan masukan telah lulus dari kedua tahap pengujian, maka akan dinyatakan bahwa bilangan tersebut adalah bilangan prima. Sebaliknya jika bilangan terserbut tidak lulus dari salah satu atau kedua tahap pengujian tersebut maka akan dinyatakan bahwa bilangan tersebut bukanlah bilangan prima.

Diagram Alir Modul

Berikut ini adalah diagram alir dari program aplikasi yang dibuat. Diagram ini terdiri dari beberapa modul yang saling berinteraksi satu sama lain sehingga dapat memproses hitungan yang ada. Adapun modul yang dimaksud adalah : modul input, modul GCD, modul faktorisasi, modul pencarian nilai m dan modul utama. Berikut ditampilkan beberapa modul

1) Diagram Alir Modul Input

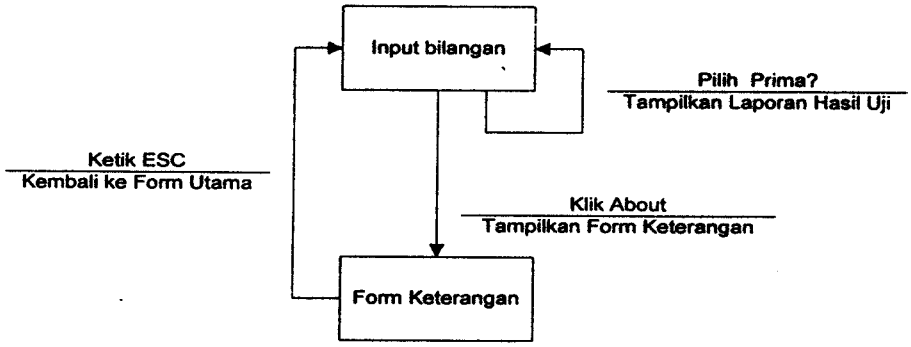
Berikut ini adalah diagram alir modul input yang menjelaskan proses validasi input



Gambar 2. Gambar Diagram Alir Modul Input

State Transition Diagram

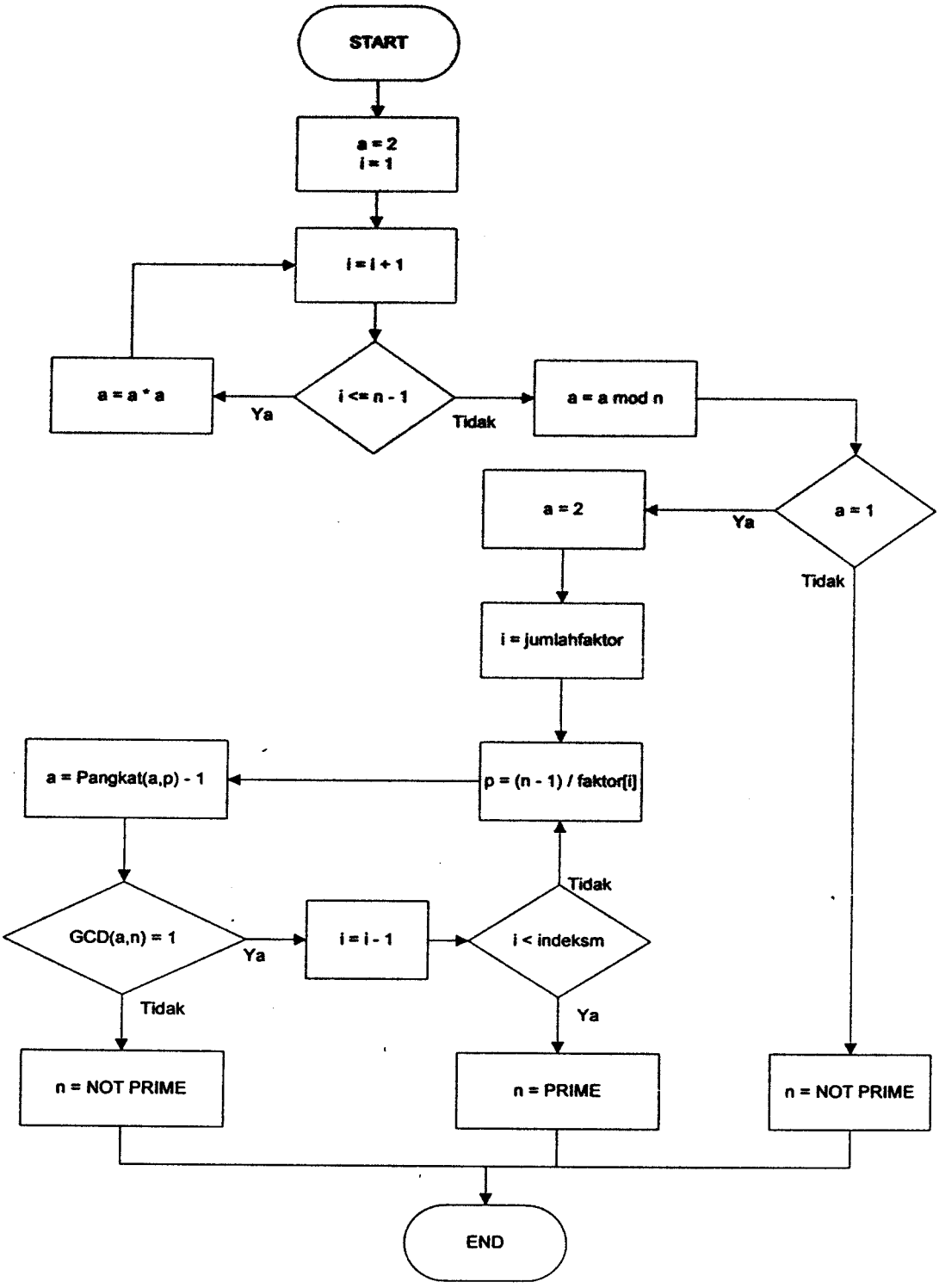
Rancangan dialog utama pada program ini dapat ditunjukkan pada gambar berikut ini.



Gambar 3. Rancangan Dialog Utama

2) Diagram Alir Modul Utama

Sedangkan digram alir program yang telah dirancang secara keseluruhan dapat ditunjukkan dalam gambar berikut.



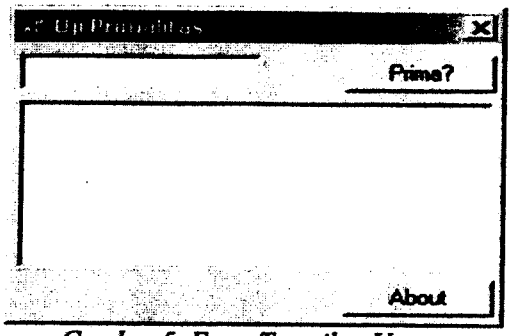
Gambar 4. Gambar Diagram Alir Modul Utama

4. Hasil dan Pembahasan

Hasil

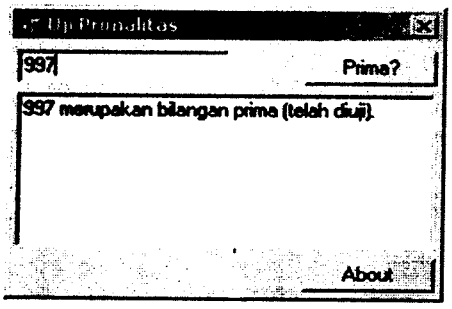
- **Modul Tampilan Utama**

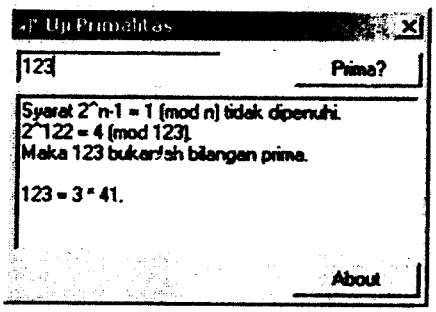
Pada saat menjalankan UjiPrima.exe, akan tampil *Form* seperti tampak pada gambar 5. Modul ini sebagai modul utama dan keseluruhan dari program.



Gambar 5. Form Tampilan Utama

Pada form ini terdapat sebuah field, di mana pemakai dapat memasukkan bilangan yang akan di uji. Tombol "Prima?" digunakan untuk memulai proses pengujian. Di bawah field input dan tombol, terdapat sebuah field *Memo*. Memo ini berfungsi untuk menampilkan laporan hasil pengujian. Jika bilangan tersebut memang benar prima, maka pada Memo akan dijelaskan bahwa bilangan tersebut telah terbukti prima. Sebaliknya, jika bukan bilangan prima, maka pada Memo akan ditampilkan salah satu faktor bilangan tersebut.





Gambar 6.a Gambar Hasil Uji Prima *Gambar 6.bGambar Hasil Uji Tidak Prima*

Pembahasan

Proses pengujian algoritma berdasarkan lamanya waktu perhitungan terhadap bilangan yang akan dicari dan pengujian algoritma uji primalitas terhadap bilangan prima .

Tabel 1. Tabel analisa lama program dalam menguji bilangan yang diinput dan hasil ujinya. Bilangan yang diinput adalah sebarang bilangan.

Bilangan	Jumlah angka	Jumlah Proses	Hasil Uji
0	1	114	Tidak Prima
1	1	114	Tidak Prima
9	1	2337	3 * 3
31	2	6024	Prima
97	2	19678	Prima
125	3	20638	5 * 125

997	3	242275	Prima
1019	4	388752	Prima
1129	4	1040062	Prima
999999	6	275032847	3 * 333333

5. Penutup

Kesimpulan

Berdasarkan data evaluasi yang diperoleh pada perancangan ini dapat disimpulkan sbb:

1. Dengan adanya program ini maka pengujian terhadap bilangan prima dapat dilakukan dan dilihat hasinya dengan cepat.
2. Semakin besar bilangan yang akan di uji, semakin lama proses pengujian.
3. Untuk bilangan – bilangan tertentu, pengujian menjadi tidak efektif, karena waktu proses perhitungan mengalami looping yg besar sehingga waktu pengujian menjadi sangat lama.
4. Sebagian besar waktu yang dibutuhkan dalam pengujian adalah pengoperasian bilangan berdasarkan tipe data yang digunakan, sehingga program tidak disarankan untuk menguji bilangan yang relatif besar.

Saran

Untuk pengembangan lebih lanjut, pada rancangan program dapat ditambahkan modul – modul baru yang mendukung proses perhitungan, yaitu algoritma konversi tipe data dan pengoperasiannya yang jauh lebih efektif sehingga waktu perhitungan dapat dipercepat dan dengan cepatnya perhitungan, pengujian terhadap bilangan yang lebih besar pun dapat dilakukan.

6. Daftar Pustaka

[1] Burton, David M., *Elementary Number Theory, Fifth Edition*, McGraw-Hill Higher Education, McGraw-Hill Company, New York, 2002.

[2] *Factorization and Primality Testing.* http://www.math.niu.edu/~rusin/known-math/index/11Y05.html, 22 Desember 2003

[3] Fitzpatrick, P. M., *Advanced Calculus*, P W S Publishing Company, Boston, 1996

[4] Jensen, Cary and Anderson, Loy, *Delphi in Depth*, McGraw-Hill Company, New York, 1999.

[5] Johnsonbaugh, Richard, *Discrete Mathematics*, Prentice Hall Inc., New Jersey, 1997.

[6] Niven, I., H. Zuckerman, and H. Montgomery, *An Introduction to the Theory of Numbers, Fifth Edition*, John Wiley & Sons Inc., New York, 1991.

[7] *Primality Tes.t* url: http://mathworld.wolfram.com/PrimalityTest.html, 10 Oktober 2003

[8] Robertson, Lesley Anne, *Simple Program Design: A Step By Step Approach, Third Edition*, Course Technology, Cambridge, 2000.

[9] Rosen, Kenneth H., *Elementary Number Theory and Its Application, Fourth Edition*, Addison Wesley Longman, Inc., Massachusetts, 2000.

[10] Stallings, William, *Cryptography and Network Security: Principles And Practice, Second Edition*, Prentice-Hall, Inc., New Jersey, 1999.

[11] Wand, M., *Induction, Recursion and Programming*, New York, 1980..